HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80527-2400

PATENT APPLICATION

ATTORNEY DOCKET NO. ___**10005459 -1**___

IN THE
UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor(s):   Robert Hundt

Application No.: 09/846222

Filing Date: Apr 30, 2001

Confirmation No.: 7243

Examiner: Qamrun Nahar

Group Art Unit: 2191

Title: Augmenting Debuggers

RECEIVED
CENTRAL FAX CENTER

MAR 1 6 2006

Mail Stop Appeal Brief-Patents
Commissioner For Patents
PO Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL OF APPEAL BRIEF

Transmitted herewith is the Appeal Brief in this application with respect to the Notice of Appeal filed on ___January 19, 2006___.

The fee for filing this Appeal Brief is (37 CFR 1.17(c)) $500.00.

(complete (a) or (b) as applicable)

The proceedings herein are for a patent application and the provisions of 37 CFR 1.136(a) apply.

☐ (a) Applicant petitions for an extension of time under 37 CFR 1.136 (fees: 37 CFR 1.17(a)-(d)) for the total number of months checked below:

☐ 1st Month $120   ☐ 2nd Month $450   ☐ 3rd Month $1020   ☐ 4th Month $1590

☐ The extension fee has already been filed in this application.

☒ (b) Applicant believes that no extension of time is required. However, this conditional petition is being made to provide for the possibility that applicant has inadvertently overlooked the need for a petition and fee for extension of time.

Please charge to Deposit Account 08-2025 the sum of ___$ 500___. At any time during the pendency of this application, please charge any fees required or credit any over payment to Deposit Account 08-2025 pursuant to 37 CFR 1.25. Additionally please charge any fees to Deposit Account 08-2025 under 37 CFR 1.16 through 1.21 inclusive, and any other sections in Title 37 of the Code of Federal Regulations that may regulate fees. A duplicate copy of this sheet is enclosed.

☐ I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to:
Commissioner for Patents, Alexandria, VA 22313-1450

Date of Deposit:

OR

☒ I hereby certify that this paper is being transmitted to the Patent and Trademark Office facsimile number (571)273-8300.

Date of facsimile: March 16, 2006   24 Pgs Faxed

Typed Name: Arla Van Limburg

Signature:

Respectfully submitted,

Robert Hundt

By_____

Tuan V. Ngo

Attorney/Agent for Applicant(s)

Reg No. :   44,259

Date :   3/16/06

Telephone :   408 447 8133

Rev 10/05 (AplBrief)

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80527-2400

Docket No.: 10005459-1

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:		
Robert Hundt		
Application No.: 09/846,222		Confirmation No.: 7243
Filed: April 30, 2001		Art Unit: 2191
For: Augmenting Debuggers		Examiner: Qamrun Nahar

APPELLANT'S BRIEF

MS Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

This brief, in compliance with 37 C.F.R. § 41.37, is in furtherance of the Notice of Appeal filed on January 19, 2006.

The fees required under § 41.20(b)(2) and any required petition for extension of time for filing this brief and fees are dealt with in the accompanying TRANSMITTAL OF APPEAL BRIEF.

03/17/2005 SSISTIX 00000063 082025 09846222

01 FC:1402 500.00 DA

Application No.: 09/846,222 Docket No.: 10005459-1

This brief contains items under the following headings as required by 37 C.F.R. § 41.37:

Page 16 of this brief bears the attorney's signature.

2

Application No.: 09/846,222 Docket No.: 10005459-1

I. REAL PARTY IN INTEREST

The real party in interest for this appeal is Hewlett-Packard Development Company, L.L.P., a Texas limited liability partnership having its principal place of business in Houston, Texas.

II. RELATED APPEALS AND INTERFERENCES

Appellant is unaware of any related appeal or interference.

III. STATUS OF CLAIMS

A. Total Number of Claims in the Application: 28 claims, which are identified as claims 1-28.

B. Current Status of Claims

1. Claims canceled: 3, 12

2. Claims withdrawn from consideration but not canceled: None

3. Claims pending: 1, 2, 4-11, 13-28

4. Claims allowed: none

5. Claims rejected: 1, 2, 4-11, 13-28

C. Claims on Appeal: 1, 2, 4-11, 13-28

IV. STATUS OF AMENDMENTS

An Amendment with all compliances ("first Amendment") in response to the first Office Action was submitted on December 16, 2004. A first Final Office Action was mailed May 6, 2005. A second Amendment in response to the first Final Office Action was mailed July 1, 2005. An Advisory was mailed July 28, 2005 and indicated that the first Final Office Action was not clear in its rejection/allowance of claims. An interview was conducted between Appellant's attorney, Tuan Ngo, and Examiner Qamrun Nahar on August 4, 2005 regarding the (un)clarity of the first Final Office Action. In the interview Examiner Nahar indicated that a Supplemental Office Action will be provided. The Interview Summary was mailed August 10, 2005. A second Final Office Action was mailed October 21, 2005 rejecting the claims in the first Amendment. A Notice of Appeal was submitted January 19, 2006 giving notice to appeal the

3

Application No.: 09/846,222   Docket No.: 10005459-1

rejections in the second Final Office Action, and this Appeal Brief is in furtherance of the Notice of Appeal. The claims on appeals are those in the first Amendment.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Claim 1 is an independent claim and recites a method for augmenting a debugger (page 5, lines 2-3) having debugging functionality used to debug a first program (page 5, line 3), comprising the steps of: providing the debugger; providing a second program (page 14, lines 8-10) having second-program functionality; and providing integration code (page 9, line 18 to page 12, line 15; page 14, line 8) for invoking a piece of code to perform a task in response to the debugging command (page 14, lines 8-10), based on types of a breakpoint (page 10, lines 8-12); wherein if the breakpoint is a debugging breakpoint, then the piece of code is selected from the debugging functionality, else if the breakpoint is an instrumentation breakpoint, then the piece of code is selected from the second-program functionality (page 10, lines 12-15).

Claim 2 is a dependent claim to independent claim 1 and recites that the method of claim 1 further comprises the step of using an instrumentor as the second program (page 14, lines 9-10).

Claim 3 has been canceled.

Claim 4 is a dependent claim to independent claim 1 and recites that the method of claim 1 further comprises the step of making the piece of code an executable part of the first program (page 8, lines 11-12; page 9, line 7).

Claim 5 is a dependent claim to independent claim 1 and recites that the method of claim 1 further comprises the step of using a trampoline as the piece of code (page 8, lines 11-12).

Claim 6 is a dependent claim to independent claim 1 and recites that the method of claim 1 wherein the debugging command is selected from one or a combination of: input from a user using the debugger (page 3, lines 5-6); a script file associated with the first program (page 12, line 13); and a configuration file associated with the first program (page 12, line 13).

Claim 7 is a dependent claim to claim 1 and recites that the method of claim 1 further comprises the step of inputting a debugging command at a debugging prompt provided by the debugger (page 3, lines 5-6; page 14, lines 4-5).

Claim 8 is a dependent claim to independent claim 1 and recites that the method of claim 1 further comprises the steps of: integrating the debugger, the second program, and the

4

Application No.: 09/846,222   Docket No.: 10005459-1

integration code into a combined code; and embedding the combined code into a language

environment (page 12, line 17 to page 13, line 15).

     Claim 9 is a dependent claim to independent claim 1 and recites that the method of claim

8 further comprises the step of using an Integrated Development Environment as the language

environment (page 13, lines 1-3).

     Claim 10 is an independent claim and recites a system (page 17, lines 7-8) for augmenting

a debugger (page 5, lines 2-3) having debugging functionality used to debug a first program

(page 5, line 3), comprising: the debugger; a second program (page 14, lines 8-10) having

second-program functionality; and integration code (page 9, line 18 to page 12, line 15; page 14,

line 8) for invoking a piece of code to perform a task in response to the debugging command

(page 14, lines 8-10), based on types of a breakpoint (page 10, lines 8-12); wherein if the

breakpoint is a debugging breakpoint, then the piece of code is selected from the debugging

functionality, else if the breakpoint is an instrumentation breakpoint, then the piece of code is

selected from the second-program functionality (page 10, lines 12-15).

     Claim 11 is a dependent claim to independent claim 10 and recites that the system of

claim 10 further comprises an instrumentor used as the second program (page 14, lines 9-10).

     Claim 12 has been canceled.

     Claim 13 is a dependent claim to independent claim 10 and recites that the system of

claim 10 wherein the piece of code is an executable part of the first program (page 8, lines 11-12;

page 9, line 7).

     Claim 14 is a dependent claim to independent claim 10 and recites that the system of

claim 10 wherein a trampoline is used as the first piece of code (page 8, lines 11-12).

     Claim 15 is a dependent claim to independent claim 10 and recites that the system of

claim 10 wherein the debugging command is selected from one or a combination of: input from a

user using the debugger (page 3, lines 5-6); a script file associated with the first program (page

12, line 13); and a configuration file associated with the first program (page 12, line 13).

     Claim 16 is a dependent claim to claim 10 and recites that the system of claim 10 wherein

the debugging command is inputted at a debugging prompt provided by the debugger (page 3,

lines 5-6; page 14, lines 4-5).

     Claim 17 is a dependent claim to independent claim 10 and recites that the system of

claim 10 wherein: the debugger, the second program, and the integration code are integrated into

5

a combined code; and the combined code is embedded in a language environment (page 12, line 17 to page 13, line 15).

Claim 18 is a dependent claim to dependent claim 17 and recites that the system of claim 17 wherein an Integrated Development Environment is used as the language environment (page 13, lines 1-3).

Claim 19 is a dependent claim and recites a computer-readable medium (page 19, lines 13-15) embodying instructions that cause a computer to perform a method for augmenting a debugger (page 5, lines 2-3) having debugging functionality used to debug a first program (page 5, line 3), the method comprising the steps of: providing the debugger; inputting a debugging command at a debugging prompt provided by the debugger (page 3, lines 5-6); providing a second program having second-program functionality; and providing integration code (page 9, line 18 to page 12, line 15; page 14, line 8) for invoking a piece of code to perform a task in response to the debugging command (page 14, lines 8-10), based on types of a breakpoint (page 10, lines 8-12); wherein if the breakpoint is a debugging breakpoint, then the piece of code is selected from the debugging functionality, else if the breakpoint is an instrumentation breakpoint, then the piece of code is selected from the second-program functionality (page 10, lines 12-15).

Claim 20 is a dependent claim to independent claim 19 and recites a computer-readable medium of claim 19 wherein the method further comprises the step of using an instrumentor as the second program (page 14, lines 9-10).

Claim 21 is a dependent claim to independent claim 1 and recites the method of claim 1 wherein the piece of code is stored in a library (page 7, lines 11-14; page 10, lines 2-3).

Claim 22 is a dependent claim to independent claim 1 and recites the method of claim 1 wherein the integration code generates the piece of code (page 11, line 18 to page 12, line 5).

Claim 23 is a dependent claim to independent claim 1 and recites the method of claim 1 wherein the integration code keeps track of modifications to the first program, and, if appropriate, undoes the modifications (page 11, lines 5-6).

Claim 24 is a dependent claim to independent claim 1 and recites the method of claim 1 wherein in a loop of more than one time, execution of code in the loop is transferred to the debugger one time (page 15, line 13 to page 16, line 1).

Claim 25 is a dependent claim to independent claim 10 and recites the system of claim 10 wherein the piece of code is stored in a library (page 7, lines 11-14; page 10, lines 2-3).

6

Application No.: 09/846,222   Docket No.: 10005459-1

     Claim 26 is a dependent claim to independent claim 10 and recites the system of claim 10 wherein the integration code generates the piece of code (page 11, line 18 to page 12, line 5).

     Claim 27 is a dependent claim to independent claim 10 and recites the system of claim 10 wherein the integration code keeps track of modifications to the first program, and, if appropriate, undoes the modifications (page 11, lines 5-6).

     Claim 28 is a dependent claim to independent claim 10 and recites the system of claim 10 wherein in a loop of more than one time, execution of code in the loop is transferred to the debugger one time (page 15, line 13 to page 16, line 1).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

     On page 3 of the second Final Office Action, paragraph 9, claims 1-2, 4-6, 10-11, 13-15, and 19-28 were rejected under 35 U.S.C. § 102(e) as being anticipated by U.S patent number 6,490,721 issued to Gorshkov (herein referred to as "Gorshkov").

     On page 9 of the second Final Office Action, paragraph 11, claims 7-9 and 16-18 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Gorshkov in view of U.S patent number issued to Leask (herein referred to as "Leask").

VII. ARGUMENT

REJECTIONS UNDER 35 U.S.C. § 102(e) – Gorshkov

     On page 3 of the second Final Office Action, paragraph 9, claims 1-2, 4-6, 10-11, 13-15, and 19-28 were rejected under 35 U.S.C. § 102(e) as being anticipated by Gorshkov. It is respectfully submitted that because Gorshkov does not teach every element of the rejected claims, these claims are patentable and the rejection should be reversed.

CLAIMS 1, 10, 19

     In response to the claimed element "a method for augmenting a debugger having debugging functionality used to debug a first program" the second Final Office Action, page 3, 4th paragraph, cited Gorshkov's paragraph of column 2, lines 58-64, which discloses a debugging subprogram and a target program to be debugged. The second Final Office Action

7

Application No.: 09/846,222 Docket No.: 10005459-1

thus corresponded Gorshkov's *debugging subprogram* and *target program* to be debugged to the *claimed debugger* and *first program* to be debugged, respectively.

In response to the claimed element "providing a second program having second-program functionality," the second Final Office Action, page 3, 5th paragraph, to page 4, 1st paragraph, asserted ". . . . Target program 10 is the existing program . . . that is to be debugged . . . . User action libraries 12 are created by compiling one or more debugging subprograms 16 . . . into linkable libraries . . . . This compiling procedure is accomplished by action compiler 14 which compiles the source code of subprograms 16 . . . . These files together with target program 10 are input to dynamic action linker 18. . . . Dynamic action linker 18 reads target program 10 and user action libraries 12 and creates two processes. A first process 20 created by dynamic action linker 18 consists of target program 10 . . . . A second process 22 created by dynamic action linker 18 handles requests from process 20 . . .; *second process 22 is interpreted as the second program having second-program functionality*" (emphasis added). Accordingly, the second Final Office Action corresponded Gorshkov's *second process 22* to the claimed *second program having second-program functionality*.

To be parallel with, and thus anticipate claim 1, Gorshkov's *second process 22* must provide functionality for the claimed piece of code when the debugging breakpoint is an instrumentation breakpoint wherein the claimed piece of code performs a task in response to a debugging command. This is because claim 1 recites ". . . invoking a piece of code to perform a task in response to a debugging command" and "if the breakpoint is an instrumentation breakpoint, then the piece of code is selected from the second-program functionality." However, Gorshkov does not disclose such feature in claim 1. That is, Gorshkov does not disclose that second process 22 provides functionality for the claimed piece of code to perform a task in response to a debugging command, if the breakpoint is an instrumentation breakpoint.

In response to the claimed element "providing integration code for invoking a piece of code to perform a task in response to a debugging command, based on types of a breakpoint; wherein if the breakpoint is a debugging breakpoint, then the piece of code is selected from the debugging functionality, else if the breakpoint is an instrumentation breakpoint, then the piece of code is selected from the second-program functionality," the second Final Office Action, page 4, 2nd paragraph to page 5, 1st paragraph, cited "[i]n step E, child process 22 has received the acknowledgment . . . and continues on to *patch* into parent process 40 a call to the dynamic user

8

Application No.: 09/846,222   Docket No.: 10005459-1

actions runtime start routines, i.e. the debugging actions. In particular, calls to user actions . . .
are inserted in the memory image of target program 10 . . . . In step F, child process waits for a
new request for *patching* service from parent process 20 . . . . Steps F and G are repeated until
the task is terminated by parent process 20 . . . . FIG. 3 illustrates Step E of FIG. 2, i.e., *patching
of dynamic user actions into target program 10 in detail.* Child process 22 . . . must *patch* the
memory image of target program 10 . . . . In step E1, child process 22 allocates space for the
patch in the *patch* area in parent process 20. In step E2, child process 22 replaces an instruction .
. . with a branch instruction to the *patch* area. In step E3, child process 22 generates code to call
the user action." in column 3, line 67 to column 4, lines 1-13; and column 4, lines 29-38"
(emphasis added).

The above cited paragraph of Gorshkov mainly discusses patching a call to user actions
runtime start routines into parent process 40, patching service from parent process 20, and
patching of dynamic user actions into target program 10, etc., and none of which is parallel to
and thus anticipates the claimed providing *an integration code for invoking a piece of code to
perform a task in response to a debugging command, based on types of a breakpoint* because
none of the patching process in Gorshkov discloses an integration code, a task to be performed in
response to a debugging command, and a piece of code to perform such task, based on types of a
breakpoint, etc.

If Gorshkov's debugging subprogram corresponds to the claimed debugger having
debugging functionality, as asserted by the second Office Action above, then to be parallel with,
and thus anticipate, the claimed element "if the breakpoint is a debugging breakpoint, then the
piece of code [to perform a task in response to a debugging command] is selected from the
debugging functionality," then Gorshkov must disclose if the breakpoint is a debugging
breakpoint, then the piece of code to perform a task in response to a debugging command is
selected from the *debugging functionality* of the *debugging subprogram.* However, Gorshkov
does not disclose such feature. Similarly, if Gorshkov's child process 22 corresponds to the
claimed second program having second-program functionality, then to be parallel with, and thus
anticipate, the claimed element "if the breakpoint is an instrumentation breakpoint, then the piece
of code [to perform a task in response to a debugging command] is selected from the second-
program functionality," then Gorshkov must disclose if the breakpoint is an instrumentation
breakpoint, then the piece of code to perform a task in response to a debugging command is

9

Application No.: 09/846,222 Docket No.: 10005459-1

selected from the *debugging functionality* of the *child process 22*. However, Gorshkov does not disclose such feature.

In any event, because Gorshkov's cited paragraph does not disclose *a debugging breakpoint* or *an instrumentation breakpoint*, Gorshkov cannot disclose selecting the piece of code from the debugging subprogram or from the process 22 if the breakpoint is a debugging breakpoint or the instrumentation breakpoint, respectively.

The second Final Office Action then continued "when the child process 22 receives a request for patching from the parent process 20, the location of the parent process 20 where the request is being made from is the location of the instrumentation breakpoint." It is respectfully submitted that a careful review of Gorshkov does not reveal that Gorshkov discloses anything to the effect that the location of the parent process 20 from where the request is being made is the location of the instrumentation breakpoint as asserted by the second Final Office Action. Even if the second Final Office Action's assertion is true, the location of the instrumentation breakpoint has no relevance to the claimed invention.

The second Final Office Action further continued "[i]f the debugging actions have already been integrated within the target program 10, the request for patching is not made. That is, the target program continues to debug using the debugging actions that has already been integrated; the debugging breakpoint has already been inserted and debugging functionality is selected." Again, a careful review of Gorshkov does not reveal anything to the effect that "the debugging breakpoint has already been inserted" as asserted. As indicated below in conjunction with claim 7, because Gorshkov is about debugging without stopping execution of the program, Gorshkov would teach away from a debugging *breakpoint*.

For the foregoing reasons, claim 1 recites limitations not disclosed in Gorshkov and is therefore patentable.

Claims 10 and 19 recite limitations corresponding to claim 1 and are therefore patentable for the same reasons as claim 1.

CLAIMS 2, 11

Claim 2, depending from claim 1, is patentable for at least the same reasons as claim 1. Even if claim 1 is not patentable, claim 2 is patentable for its additional limitation of "using an instrumentor as the second program."

10

Gorhskov's cited paragraph of col. 4, lines 29-38 (second Final Office Action, page 5, middle of the page) discloses patching of dynamic users actions into target program 10 or patching dynamic user actions into the claimed first program as the claimed first program was corresponded to target program 10 (above, in the discussion of claim1). This cited paragraph discloses that child process 22 allocates space for the patch and replaces instructions (col. 4, lines 33-37) at the requested program location or the claimed second program allocates space for the patch and replaces instructions at the requested program location, but does not disclose the claimed instrumentor or using this claimed instrumentor as Gorshkov's child process 22, which was corresponded to the claimed second program (above, in the discussion of claim 1).

Claim 11 recites limitations corresponding to claim 2 and is therefore patentable for the same reasons as claim 2.

CLAIM 3

Claim 3 has been canceled and is not the subject for this appeal.

CLAIMS 4, 13

Claim 4, depending from claim 1, is patentable for at least the same reasons as claim 1. Even if claim 1 is not patentable, claim 4 is patentable for its additional limitation of making the piece of code an executable part of the first program. The claimed piece of code is patentably distinguished from Gorshkov's cited debugging actions (second Final Office Action, page 6, first paragraph) because these actions *are not invoked by an integration code in response to the debugging command* as in claim 4.

Claim 13 recites limitations corresponding to claim 4 and is therefore patentable for the same reasons as claim 4.

CLAIMS 5, 14

Claim 5, depending from claim 1, is patentable for at least the same reasons as claim 1. Even if claim 1 is not patentable, claim 5 is patentable for its additional limitation of using a trampoline as the piece of code. The second Final Office Action was contradictory when it corresponded child *process 22 to the claimed piece of code* (second Final Office Action, page 6, middle of the page) because *child process 22 had been corresponded to the claimed second*

11

Application No.: 09/846,222   Docket No.: 10005459-1

program. There is not any piece of code in Gorshkov that is patentably parallel to the claimed piece of code which was invoked by the integration code in response to a debugging command. There is not any disclose in Gorshkov that using a trampoline as the piece of code.

Claim 14 recites limitations corresponding to claim 5 and is therefore patentable for the same reasons as claim 5.

CLAIMS 6, 15

Claim 6, depending from claim 1, is patentable for at least the same reasons as claim 1. Even if claim 1 is not patentable, claim 6 is patentable for its additional limitations as recited. There is not anything in Gorshkov that is patentably parallel to the claimed debugging command because this command may be in response to an instrumentation breakpoint, which is clearly not disclosed in Gorshkov.

Claim 15 recites limitations corresponding to claim 6 and is therefore patentable for the same reasons as claim 6.

CLAIMS 21, 25

Claim 21, depending from claim 1, is patentable for at least the same reasons as claim 1. Even if claim 1 is not patentable, claim 21 is patentable for its additional limitation that the claimed piece of code is stored in a library. There is not anything in Gorshkov that is patentably parallel to the claimed piece of code because the claimed piece of code is invoked from an integration code that is not disclosed in Gorshkov. Therefore, there cannot be any disclosure that the claimed piece of code is stored in a library.

Claim 25 recites limitations corresponding to claim 21 and is therefore patentable for the same reasons as claim 21.

CLAIMS 22, 26

Claim 22, depending from claim 1, is patentable for at least the same reasons as claim 1. Even if claim 1 is not patentable, claim 22 is patentable for its additional limitation that the integration code generates the piece of code. As discussed above, there is not anything in Gorshkov that is patentably parallel to the claimed piece of code. Further, there is not anything in Gorshkov that is patentably parallel to the claimed integration code. The claimed integration

12

Application No.: 09/846,222   Docket No.: 10005459-1

code, illustratively disclosed in the Specification, page 9, line 18 to page 12, line 15, is

patentably distinguished from the cited user actions, user action libraries 12, etc. (col. 4, lines 4-

8, second Final Office Action, page 7). Therefore, there cannot be any disclosure that the

claimed instrumentation code generates the claimed piece of code.

Claim 26 recites limitations corresponding to claim 22 and is therefore patentable for the

same reasons as claim 22.

CLAIMS 23, 27

Claim 23, depending from claim 1, is patentable for at least the same reasons as claim 1.

Even if claim 1 is not patentable, claim 23 is patentable for its additional limitation that the

integration code keeps track of modifications to the first program, and, if appropriate, undoes the

modifications. Gorshkov's cited paragraph of col. 4, lines 48-53 discloses that *the code in the*

patch area restores saved registers from the patch stack, returns the space allocated from the

patch stack, executes the instructions that were removed from the subroutine entry sequence, and

return to subroutine. The remaining code is then executed and execution is returned to the caller

of the subroutine. Here, there is not any disclosure of any code that is patentaby parallel to the

claimed integration code, the integration code that keeps track of the Gorshkov's target program

to be debugged, and that the integration code undoes the modifications as appropriate. The

claimed integration code is patentably distinguished from Gorshkov's code in the patch area for

at least the reason that the claimed integration code invokes a piece of code in response to the

debugging command based on types of a breakpoint while there is not any disclosure to that

effect for Gorshkov's code in the patch area.

Claim 27 recites limitations corresponding to claim 23 and is therefore patentable for the

same reasons as claim 23.

CLAIMS 24, 28

Claim 24, depending from claim 1, is patentable for at least the same reasons as claim 1.

Even if claim 1 is not patentable, claim 24 is patentable for its additional limitation that in a loop

of more than one time, execution of code in the loop is transferred to the debugger one time.

Gorshkov's cited paragraph of col. 4, lines 57-65 discloses the calling of a subroutine, branching

into the patch area, the code in the patch area that is supposed to perform a certain task, etc.

13

Application No.: 09/846,222 Docket No.: 10005459-1

However, there is not any disclosure about execution of a loop, transferring execution of the code in the loop to Gorhkov's debugging program (corresponded to the claimed debugger) or execution of code in the loop is executed one time in a loop of more than one time.

Claim 28 recites limitations corresponding to claim 24 and is therefore patentable for the same reasons as claim 24.

REJECTIONS UNDER 35 U.S.C. § 103(a) – Gorshkov and Leask

On page 9 of the second Final Office Action, paragraph 11, claims 7-9 and 16-18 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Gorshkov in view of "Leask" (U.S. patent number 6,412,106). It is respectfully submitted that Gorshkov and Leask, either alone or in combination, do not teach every element of the claimed inventions. The alleged motivation for combining Gorshkov and Leask is improper. Showing of a prima facie case of obviousness fails.

Leask does not provide the missing elements in Gorshkov that were discussed above in conjunction with claim 1. Claims 7-9, depending from claim 1, are patentable for at least the same reasons as claim 1. Claims 7-9 are also patentable for their additional limitations. Claims 16-18, depending from claim 10, are patentable for at least the same reasons as claim 10. Claims 16-18 are also patentable for their additional limitations.

CLAIMS 7, 16

Regarding claim 7, both Gorshkov and Leask teach away from inputting a debugging command at a *debugging breakpoint* provided by the debugger because it is *an object* of Gorshkov to permit "debugging of a computer program *without stopping* execution of the program" (col. 2, lines 21-23, emphasis added) and "*noninteractive* debugging of a computer program" (col. 2, lines 24-25, emphasis added) while Leask teaches "a developer may effectively debug an application program without being required *to halt the program* by . . . inserting breakpoints while the program is running" (col. 7, lines 62-65, emphasis added). Therefore, there is not any motivation for combining Gorshkov and Leask.

Claim 16 recites limitations corresponding to claim 7 and is therefore patentable for the same reasons as claim 7.

14

Application No.: 09/846,222 Docket No.: 10005459-1

CLAIMS 8, 17

Regarding claim8, Leask's cited paragraph of col. 12, lines 18-36 (second Fianl Office Action, page 9, last paragraph) discloses that the graphical debugging environment is capable of displaying a graphical representation of an application program. Additionally, various testing and debugging tools may be offered to a developer in a graphical debugging environment. As can be seen, there is not any disclosure about the claimed debugger, the claimed second program, and the claimed integration code. As a result, there cannot be any disclosure about combining the claimed debugger, the claimed second program, and the claim integration code and embedding them into a language environment as in claim 8

The alleged motivation for combining Leask into Gorshkov is improper as the second Final Office Action failed to show any teaching or suggestion in either Leasks or Gorshkov to combine the two teachings of Gorshkov or it is of common knowledge to combine the two teachings.

Claim 17 recites limitations corresponding to claim 8 and is therefore patentable for the same reasons as claim 8.

CLAIMS 9, 18

Regarding claim 9, the rejection of which is based on Leask's same paragraph of col. 12, lines 18-36 used to reject claim 8. As discussed above in relation to claim 8, the cited paragraph does not disclose the claimed debugger, the claimed second program, and the claimed integration code. As a result, there cannot be any disclosure about combining the claimed debugger, the claimed second program, and the claim integration code and embedding them into an Integrated Development Environment.

Claim 18 recites limitations corresponding to claim 9 and is therefore patentable for the same reasons as claim 9.

15

Application No.: 09/846,222 Docket No.: 10005459-1

CONCLUSIONS
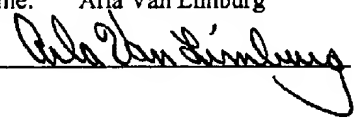
For the forgoing reasons, pending claims 1-2, 4-11, 13-28 are patentably distinguished from the prior art of record and are therefore patentable. Reversal of all claim rejections is respectfully solicited.

I hereby certify that this paper is being transmitted to the Patent and Trademark Office facsimile number (571) 273-8300.

Date of facsimile: March 16, 2006

Number of Pages: 24

Typed Name: Arla Van Limburg

Signature: _____

Respectfully Submitted,

By: _____

Tuan V. Ngo
Attorney for Appellant
Reg. No. 44,259

Date: 3/16/06

Telephone No. (408) 447-8133

16

Application No.: 09/846,222   Docket No.: 10005459-1

VIII. CLAIMS APPENDIX

Claims Involved in the Appeal of Application Serial No. 09/846,222

1   1. (Previously Presented) A method for augmenting a debugger having debugging

2   functionality used to debug a first program, comprising the steps of:

3   providing the debugger;

4   providing a second program having second-program functionality; and

5   providing integration code for invoking a piece of code to perform a task in

6   response to a debugging command, based on types of a breakpoint;

7   wherein if the breakpoint is a debugging breakpoint, then the piece of code is

8   selected from the debugging functionality, else if the breakpoint is an

9   instrumentation breakpoint, then the piece of code is selected from the

10   second-program functionality.

1   2. (Original) The method of claim 1 further comprises the step of using an instrumentor as the

2   second program.

1   3. (Canceled)

1   4. (Previously Presented) The method of claim 1 further comprises the step of making the

2   piece of code an executable part of the first program.

1   5. (Previously Presented) The method of claim 1 further comprises the step of using a

2   trampoline as the piece of code.

1   6. (Previously Presented) The method of claim 1 wherein the debugging command is selected

2   from one or a combination of:

3   input from a user using the debugger;

17

Application No.: 09/846,222   Docket No.: 10005459-1

4   a script file associated with the first program; and

5   a configuration file associated with the first program.

1   7. (Previously Presented) The method of claim 1 further comprises the step of inputting the

2   debugging command at a debugging prompt provided by the debugger.

1   8. (Previously Presented) The method of claim 1 further comprises the steps of:

2   integrating the debugger, the second program, and the integration code into a

3   combined code; and

4   embedding the combined code into a language environment.

1   9. (Previously Presented) The method of claim 8 further comprises the step of using an

2   Integrated Development Environment as the language environment.

1   10. (Previously Presented) A system for augmenting a debugger having debugging

2   functionality used to debug a first program, comprising:

3   the debugger;

4   a second program having second-program functionality; and

5   integration code for invoking a piece of code to perform a task in response to a

6   debugging command, based on types of a breakpoint;

7   wherein if the breakpoint is a debugging breakpoint, then the piece of code is

8   selected from the debugging functionality, else if the breakpoint is an

9   instrumentation breakpoint, then the piece of code is selected from the

10   second-program functionality.

1   11. (Original) The system of claim 10 further comprises an instrumentor used as the second

2   program.

1   12. (Canceled)

18

.1   13. (Previously Presented) The system of claim 10 wherein the piece of code is an executable

2   part of the first program.

1   14. (Previously Presented) The system of claim 10 wherein a trampoline is used as the first

2   piece of code.

1   15. (Previously Presented) The system of claim 10 wherein the debugging command is

2   selected from one or a combination of:

3   input from a user using the debugger;

4   a script file associated with the first program; and

5   a configuration file associated with the first program.

1   16. (Previously Presented) The system of claim 10 wherein the debugging command is

2   inputted at a debugging prompt provided by the debugger.

1   17. (Previously Presented) The system of claim 10 wherein:

2   the debugger, the second program, and the integration code are integrated into

3   a combined code; and

4   the combined code is embedded in a language environment.

1   18. (Previously Presented) The system of claim 17 wherein an Integrated Development

2   Environment is used as the language environment.

1   19. (Previously Presented) A computer-readable medium embodying instructions that cause a

2   computer to perform a method for augmenting a debugger having debugging

3   functionality used to debug a first program, the method comprising the steps of:

4   providing the debugger;

5   providing a second program having second-program functionality; and

19

Application No.: 09/846,222   Docket No.: 10005459-1

6   providing integration code for invoking a piece of code to perform a task in

7   response to a debugging command, based on types of a breakpoint;

8   wherein if the breakpoint is a debugging breakpoint, then the piece of code is

9   selected from the debugging functionality, else if the breakpoint is an

10   instrumentation breakpoint, then the piece of code is selected from the

11   second-program functionality.

1   20. (Original) The computer-readable medium of claim 19 wherein the method further

2   comprises the step of using an instrumentor as the second program.

1   21. (Previously Presented) The method of claim 1 wherein the piece of code is stored in a

2   library.

1   22. (Previously Presented) The method of claim 1 wherein the integration code generates the

2   piece of code.

1   23. (Previously Presented) The method of claim 1 wherein the integration code keeps track of

2   modifications to the first program, and, if appropriate, undoes the modifications.

1   24. (Previously Presented) The method of claim 1 wherein in a loop of more than one time,

2   execution of code in the loop is transferred to the debugger one time.

1   25. (Previously Presented) The system of claim 10 wherein the piece of code is stored in a

2   library.

1   26. (Previously Presented) The system of claim 10 wherein the integration code generates the

2   piece of code.

Application No.: 09/846,222   Docket No.: 10005459-1

1   27. (Previously Presented) The system of claim 10 wherein the integration code keeps track
2   of modifications to the first program, and, if appropriate, undoes the modifications.

1   28. (Previously Presented) The system of claim 10 wherein in a loop of more than one time,
2   execution of code in the loop is transferred to the debugger one time.

21

Application No.: 09/846,222   Docket No.: 10005459-1

IX.   EVIDENCE APPENDIX

No evidence was submitted pursuant 37 C.F.R. §§ 1.130, 1.131, and/or 1.132, nor was evidence entered by the Examiner and relied upon by the Appellant in the appeal.

X.   RELATED PROCEEDINGS APPENDIX

Appellant is unaware of any related proceedings, including appeals or interferences.

22